

Package: jpaccidents (via r-universe)

October 18, 2026

Title Tools for Traffic Accident Data in Japan

Version 0.2.6

Description In Japan, traffic accident data are published by the National Police Agency. This package provides tools to efficiently handle and standardise these datasets, facilitating seamless analysis and integration into various workflows. By using this package, users can uncover patterns in traffic accidents and identify high-risk scenarios, supporting efforts to enhance road safety and prevent future incidents.

License MIT + file LICENSE

URL <https://nononoexe.github.io/jpaccidents/>

Depends R (>= 4.0)

Imports cli, curl, dplyr, fs, lubridate, magrittr, purrr, readr, rlang, sf, stringr, tibble, tidyr

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

Encoding UTF-8

LazyData true

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.3

Config/pak/sysreqs libabsl-dev cmake libgdal-dev gdal-bin libgeos-dev make libicu-dev libuv1-dev libssl-dev libproj-dev libsqlite3-dev libudunits2-dev libx11-dev

Repository <https://nononoexe.r-universe.dev>

Date/Publication 2026-01-20 02:45:18 UTC

RemoteUrl <https://github.com/NONONOexe/jpaccidents>

RemoteRef HEAD

RemoteSha 2d764c4743455d47f5e6804b64032de698e665a4

Contents

download_accident_data	2
filter_accident_data	3
filter_collision_pattern	4
merge_accident_data	5
read_accident_data	6
Index	7

download_accident_data
<i>Download the traffic accident data</i>

Description

download_accident_data() downloads the file of traffic accident data provided the **National Police Agency**.

Usage

download_accident_data(type, download_dir = getwd(), years = 2023)

Arguments

type	A character string specifying the type of data to download: one of "main", "supp", or "highway".
download_dir	A character string specifying the directory to save the downloaded files (default: getwd()).
years	An integer or vector of integers specifying the years of the data to download. Available years are 2019 to 2023 (default: 2023).

Details

The traffic accident data are divided into the following types:

- "main" : Basic data on traffic accidents (honhyo).
- "supp" : Additional data related to the "main" data (hojuhyo).
- "highway" : Data on traffic accidents on highways (kosokuhyo).

Use the type argument to specify which data type to download.

Value

A character vector of file paths for the downloaded files (invisibly).

Examples

```
## Not run:
# Download the main data for 2023
download_accident_data("main", "download-dir-path", 2023)

# Download the supplementary data for 2023
download_accident_data("supp", "download-dir-path", 2023)

# Download the highway data for 2019 to 2023
download_accident_data("highway", "download-dir-path", 2019:2023)

## End(Not run)
```

filter_accident_data *Filter accident data by categories*

Description

These functions filter accident data based on the specified categories.

Usage

```
filter_accident_type(data, accident_type)
```

```
filter_road_shape(data, road_shape)
```

Arguments

data	A accident_data object.
accident_type	A vector of specifying the accident type to filter. Must be one of the following values: • "vehicle_to_pedestrian": Vehicle-to-pedestrian collision • "vehicle_to_vehicle": Vehicle-to-vehicle collision • "single_vehicle": Single-vehicle accident • "train": Train accident
road_shape	A vector of specifying the road type to filter. Must be one of the following values: • "intersection_roundabout": Roundabout intersection • "intersection_other": Regular intersection • "near_intersection_roundabout": Area near a roundabout • "near_intersection_other": Area near an intersection • "tunnel": Tunnel • "bridge": Bridge • "curve": Curve or bend in the road • "single_road": Straight single road

- "railroad_crossing_type1" : Type 1 railroad crossing
- "railroad_crossing_type3" : Type 3 railroad crossing
- "railroad_crossing_type4" : Type 4 railroad crossing
- "general_traffic_area" : General traffic area

Value

A filtered accident_data object.

Examples

```
## Not run:
# Extract accidents where vehicle collided with pedestrian
vehicle_to_pedestrian_collisions <- data %>%
  filter_accident_type("vehicle_to_pedestrian")

# Extract accidents involving vehicle-to-vehicle collisions
vehicle_to_pedestrian_collisions <- data %>%
  filter_accident_type("vehicle_to_pedestrian")

# Extract accidents that occurred on curved roads
curved_road_accidents <- data %>%
  filter_road_shape("curve")

## End(Not run)
```

filter_collision_pattern

Filter accident data by collision patterns

Description

filter_collision_pattern() filters accident data based on primary and secondary impact conditions for each party involved in the accident.

Usage

```
filter_collision_pattern(
  data,
  primary_impacts,
  secondary_impacts = NA_character_
)
```

Arguments

data A list containing accident data.

primary_impacts A vector specifying the initial collision points (primary impact) for each involved party.

secondary_impacts

A vector specifying the subsequent collision points (secondary impact) for each involved party (default: NA_character_).

Details

The values of primary_impacts and secondary_impacts are following:

- "no_impact" : No impact
- "front" : Front of the vehicle
- "right" : Right side of the vehicle
- "rear" : Rear of the vehicle
- "left" : Left side of the vehicle
- "front_right" : Front right of the vehicle
- "rear_right" : Rear right of the vehicle
- "rear_left" : Rear left of the vehicle
- "front_left" : Front left of the vehicle The order of values in primary_impacts and secondary_impacts corresponds to the party order (party_order).

Value

A list containing filtered accident data.

Examples

```
## Not run:
# Filter accident data by cases indicating head-on collisions
head_on_collisions <- filter_collision_pattern(data, c("front", "front"))

# Filter accident data by cases indicating rear-end collisions
rear_end_collisions <- filter_collision_pattern(data, c("rear", "front"))

## End(Not run)
```

merge_accident_data	<i>Merge accident data</i>
---------------------	----------------------------

Description

merge_accident_data() merges multiple accident_data objects into a single accident_data object.

Usage

```
merge_accident_data(...)

## S3 method for class 'accident_data'
merge_accident_data(...)

## S3 method for class 'list'
merge_accident_data(data_list, ...)
```

Arguments

```
...      accident_data objects to merge.
data_list  A list of accident_data objects to merge.
```

Value

A merged accident_data object.

read_accident_data	<i>Read accident data from CSV files</i>
--------------------	--

Description

read_accident_data() reads accident data from multiple CSV files, processes each file, and combines the data into three categories:

- `accident_info`: Basic accident information including date, location, and conditions
- `person_info`: Information about individuals involved in the accidents
- `highway_info`: Information specific to accidents that occurred on highways/expressways

Usage

```
read_accident_data(file_paths)
```

Arguments

```
file_paths      A character vector of file paths to the CSV files.
```

Value

A list containing three data frames (`accident_info`, `person_info`, and `highway_info`).

Examples

```
## Not run:
# Read main and supplementary data for 2023
read_accident_data(c("example/honhyo_2023.csv", "example/hojuhyo_2023.csv"))

## End(Not run)
```

Index

`download_accident_data`, [2](#)

`filter_accident_data`, [3](#)

`filter_accident_type`
 (`filter_accident_data`), [3](#)

`filter_collision_pattern`, [4](#)

`filter_road_shape`
 (`filter_accident_data`), [3](#)

`merge_accident_data`, [5](#)

`read_accident_data`, [6](#)